In the Claims:

Please amend Claims 1, 11, 21, 31-34, 37 and 40, all as shown below. Applicant respectfully reserves the right to prosecute any originally presented claims in a continuing or future application.

1. (Currently Amended) A system for organization of software application files during development and subsequent deployment of the software application to a server, comprising:

a split directory structure stored on a computer medium that stores files for a software application, wherein, for each software application, the split directory structure includes both a source folder that stores editable source files as part of the software application, and a corresponding output folder that stores system-generated compiled ~~files~~ versions of the source files as part of the software application, together with a build file that indicates the output folder as being an output of its corresponding source folder, and wherein the split directory is accessed as a virtual file that provides an abstraction over the two folders therein, so that the two folders appear to a server as a single application;

a server upon which the software application will be deployed; and

a deployment tool that allows a user to specify the output folder during deployment of the software application, wherein during the deployment of an application class or resource the server interprets the software application as a union of both the source folder and the output folder, recognizes the split directory structure, parses the build file to determine the corresponding source folder, and deploys the application by making requests to the virtual file which checks first the source folder for the class or resource, and then the corresponding output folder for ~~software application files~~ that class or resource, before deploying the software application files to the server.

2-3. (Canceled).

4. (Previously Presented) The system of claim 1 wherein the output folder includes a file that identifies the output folder as being part of the split directory which also includes the corresponding source folder.

5. (Canceled).

6. (Original) The system of claim 1 wherein said software application, or another software application can point to the output folder to access or retrieve resources in either the output folder and/or the source folder as necessary for operation of the software application.

7. (Original) The system of claim 1 wherein said output folder is automatically created and populated upon compiling the software application.

8. (Original) The system of claim 1 wherein said output folder can be deleted to remove the latest build of the software application, and then recreated to create a new build.

9. (Canceled).

10. (Original) The system of claim 1 wherein the source folder is populated with source files that are stored in or retrieved from a source control system.

11. (Currently Amended) A method for deploying a software application to a server, comprising the steps of:

storing files for a software application in a split directory structure on a computer medium, wherein, for each software application, the split directory structure includes both a source folder that stores editable source files as part of the software application, and a corresponding output folder that stores system-generated compiled ~~files~~ versions of the source files as part of the software application, together with a build file that indicates the output folder as being an output of its corresponding source folder, and wherein the split directory is accessed as a virtual file that provides an abstraction over the two folders therein, so that the two folders appear to a server as a single application; and

allowing the user to specify the output folder during deployment of an application class or resource of the software application to the server;

wherein during the deployment the server

interprets the software application as a union of both the source folder and the output folder, parses the build file to determine the corresponding source folder, recognizes the split directory structure by making requests to the virtual file which checks first the source folder for the class or resource, and then the corresponding output folder for ~~software application files~~ that class or resource, and

deploys the software application files to the server.

12-13. (Canceled).

14. (Previously Presented) The method of claim 11 wherein the output folder includes a file that identifies the output folder as being part of the split directory which also includes the corresponding source folder.

15. (Canceled).

16. (Original) The method of claim 11 wherein said software application, or another software application can point to the output folder to access or retrieve resources in either the output folder and/or the source folder as necessary for operation of the software application.

17. (Original) The method of claim 11 wherein said output folder is automatically created and populated upon compiling the software application.

18. (Original) The method of claim 11 wherein said output folder can be deleted to remove the latest build of the software application, and then recreated to create a new build.

19. (Canceled).

20. (Original) The method of claim 11 wherein the source folder is populated with source files that are stored in or retrieved from a source control system.

21. (Currently Amended) A computer readable medium including instructions stored thereon which when executed cause the computer to perform the steps of:

storing files for a software application in a split directory structure on a computer medium, wherein, for each software application, the split directory structure includes both a source folder that stores editable source files as part of the software application, and a corresponding output folder that stores system-generated compiled ~~files~~ versions of the source files as part of the software application, together with a build file that indicates the output folder as being an output of its corresponding source folder, and wherein the split directory is accessed as a virtual file that provides an abstraction over the two folders therein, so that the two folders appear to a server as a single application;

allowing the user to specify the output folder during deployment of an application class or resource of the software application to the server;

interpreting the software application as a union of both the source folder and the output folder, parsing the build file to determine the corresponding source folder, and recognizing the split directory structure by making requests to the virtual file which checks first the source folder for the class or resource, and then the corresponding output folder for ~~software application files~~ that class or resource; and

deploying the software application files to the server.

22-23. (Canceled).

24. (Previously Presented) The computer readable medium of claim 21 wherein the output folder includes a file that identifies the output folder as being part of the split directory which also includes the corresponding source folder.

25. (Canceled).

26. (Original) The computer readable medium of claim 21 wherein said software application, or another software application can point to the output folder to access or retrieve resources in

either the output folder and/or the source folder as necessary for operation of the software application.

27. (Original) The computer readable medium of claim 21 wherein said output folder is automatically created and populated upon compiling the software application.

28. (Original) The computer readable medium of claim 21 wherein said output folder can be deleted to remove the latest build of the software application, and then recreated to create a new build.

29. (Canceled).

30. (Original) The computer readable medium of claim 21 wherein the source folder is populated with source files that are stored in or retrieved from a source control system.

31. (Currently Amended) The system of claim 1 wherein the virtual file first checks the source folder for the software application files including any classes or resources needed by the software application, and, if the classes or resources are not found in the source folder, then checks the output ~~directory~~ folder.

32. (Currently Amended) The method of claim 11 wherein the virtual file first checks the source folder for the software application files including any classes or resources needed by the software application, and, if the classes or resources are not found in the source folder, then checks the output ~~directory~~ folder.

33. (Currently Amended) The computer readable medium of claim 21 wherein the virtual file first checks the source folder for the software application files including any classes or resources needed by the software application, and, if the classes or resources are not found in the source folder, then checks the output ~~directory~~ folder.

34. (Currently Amended) The system of claim 1 wherein the source folder includes ~~Java~~ JAVA files from a source control system, and wherein the output folder includes generated ~~Java~~ JAVA class files, and wherein when the server searches for a resource or class for use in deploying the software application, it first checks the source folder, and if the class or resource is not found, then checks the output ~~directory~~ folder.

35. (Previously Presented) The system of claim 1 wherein the system comprises a different split directory structure and corresponding virtual file for each software application to be deployed to the server.

36. (Previously Presented) The system of claim 1 wherein source files can be modified by the user and then stored in the source folder, and wherein the server automatically sees the new source files and uses them as necessary in deploying the software application.

37. (Currently Amended) The method of claim 11 wherein the source folder includes ~~Java~~ JAVA files from a source control system, and wherein the output folder includes generated ~~Java~~ JAVA class files, and wherein when the server searches for a resource or class for use in deploying the software application, it first checks the source folder, and if the class or resource is not found, then checks the output folder.

38. (Previously Presented) The method of claim 11 wherein the method comprises using a different split directory structure and corresponding virtual file for each software application to be deployed to the server.

39. (Previously Presented) The method of claim 11 wherein source files can be modified by the user and then stored in the source folder, and wherein the server automatically sees the new source files and uses them as necessary in deploying the software application.

40. (Currently Amended) The computer readable medium of claim 21 wherein the source folder includes ~~Java~~ JAVA files from a source control system, and wherein the output folder includes

generated ~~Java~~ JAVA class files, and wherein when the server searches for a resource or class for use in deploying the software application, it first checks the source folder, and if the class or resource is not found, then checks the output folder.

41. (Previously Presented) The computer readable medium of claim 21 wherein the steps comprise using a different split directory structure and corresponding virtual file for each software application to be deployed to the server.

42. (Previously Presented) The computer readable medium of claim 21 wherein source files can be modified by the user and then stored in the source folder, and wherein the server automatically sees the new source files and uses them as necessary in deploying the software application.